|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | |  |  |  | | МИНОБРНАУКИ РОССИИ | | | | Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА** – **Российский технологический университет»**  **РТУ МИРЭА** | | | | |
| Институт ИКБ | |
| Кафедра защиты информации | |

|  |  |
| --- | --- |
| **ОТЧЁТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ № 3** | |
| **по дисциплине «Технологии и методы программирования»** | |
|  | |
| Выполнила студент группы БИСО-02-21 | Одинцова Ю.А. |
| Проверил | Лесько С.А. |

Москва 2023

**Практическая работа №3**

**Стратегия**

Ссылка:

<https://github.com/just-student-19/tmp_6_sem>

Стратегия — это поведенческий паттерн проектирования, который определяет семейство схожих алгоритмов и помещает каждый из них в собственный класс, после чего алгоритмы можно взаимозаменять прямо во время исполнения программы.

*Листинг программы:*

class IStrategy {

execute() {

throw new Error('Unknown error!');

}

}

class Strategy1 extends IStrategy {

execute() {

console.log('Следуем стратегии №1');

}

}

class Strategy2 extends IStrategy {

execute() {

console.log('Следуем стратегии №2');

}

}

class Context {

constructor(strategy) {

this.\_strategy = strategy;

}

setStrategy(newStrategy) {

this.\_strategy = newStrategy;

}

executeStrategy() {

this.\_strategy.execute();

}

}

// Main

const strategy1 = new Strategy1();

const strategy2 = new Strategy2();

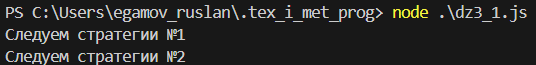
const context = new Context(strategy1);

context.executeStrategy();

context.setStrategy(strategy2);

context.executeStrategy();

*Скриншот выполнения программы:*



# Шаблонный метод

Шаблонный метод — это поведенческий паттерн проектирования, который определяет скелет алгоритма, перекладывая ответственность за некоторые его шаги на подклассы. Паттерн позволяет подклассам переопределять шаги алгоритма, не меняя его общей структуры

*Листинг программы:*

class AbstractClass {

templateMethod() {

this.step1();

this.step2();

this.step3();

}

step1() {

throw new Error('Ошибка 1');

}

step2() {

throw new Error('Ошибка 2');

}

step3() {

throw new Error('Ошибка 3');

}

}

class TemplateMethodClass extends AbstractClass {

step1() {

console.log('Шаг 1 выполнен');

}

step2() {

console.log('Шаг 2 выполнен');

}

step3() {

console.log('Шаг 3 выполнен');

}

}

// Main

const templateMethod = new TemplateMethodClass();

templateMethod.templateMethod();

*Скриншот выполнения программы:*

